Mixture models for continuous data

Generalized Processing Tree Models

Daniel W. Heck

Statistical Modeling in Psychology
FREIBURG HEIDELBERG LANDAU MANNHEIM TÜBINGEN

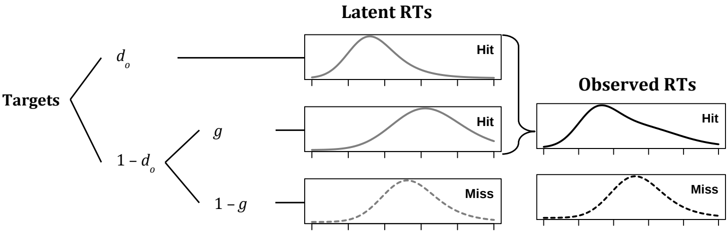
2018-09-11

MPT Modeling with Continuous Data

1. MPT-RT: Modeling response times with histograms
   - Heck & Erdfelder (2016)
2. GPT (generalized processing tree): Parametric modeling
   - Heck, Erdfelder, & Kieslich (in press)
3. RT-MPT: Serial-process model for response times
   - Klauer & Kellen (2018)

Mixture distribution

- All of the MPT extensions assume mixture distributions for discrete and continuous observations
  1. Latent RTs: Different processing branches of the MPT model result in different latent distributions $g_j(t)$
  2. Observed RTs: A mixture distribution, defined as $f(t) = \sum_j p_j g_j(t)$
  3. The mixture weights p_j are determined by the MPT structure (= branch probabilities)

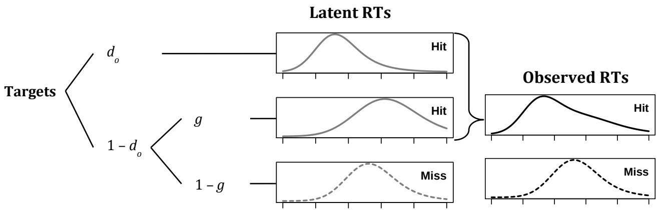


Generalized Processing Tree Models

- Heck, D. W., Erdfelder, E., & Kieslich, P. J. (in press). Generalized processing tree models: Jointly modeling discrete and continuous variables. *Psychometrika.*

Generalized processing tree (GPT) models

- Main difference: Parametric assumptions for component distributions
- The type of distribution depends on continuous variable
  - RTs: log-normal, ex-Gaussian, ...
  - Mouse-tracking measures (see below): Normal distribution
  - Neuro-psychological measures: ...
- The distributions are described by parameters $\boldsymbol{\eta}$
  - Normal distribution: mean and SD
  - ex-Gaussian: mean, SD, and mean of exponential



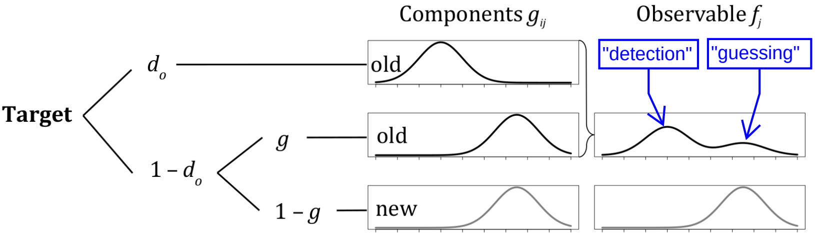
Generalized Processing Tree (GPT) Models

Benefits of the GPT Framework

1. Increased precision in estimating MPT parameters $\boldsymbol{\theta}$
2. Unidentifiable MPT models can become identifiable
3. Flexibility and simplicity

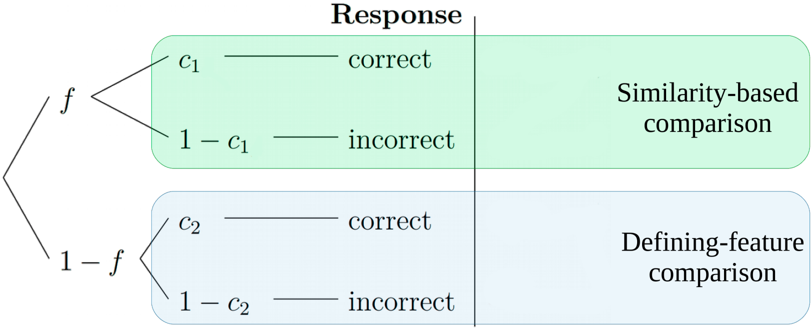
Higher precision of MPT-parameter estimates in GPTs

- The more distinct the latent distributions, the smaller the standard error
- Intuition: Continuous variables improve the “classification” which trials belong to which latent processing states
- 2HTM: “Fast RTs are due to detection, slow RTs are due to guessing”



Example: The feature comparison model of semantic categorization

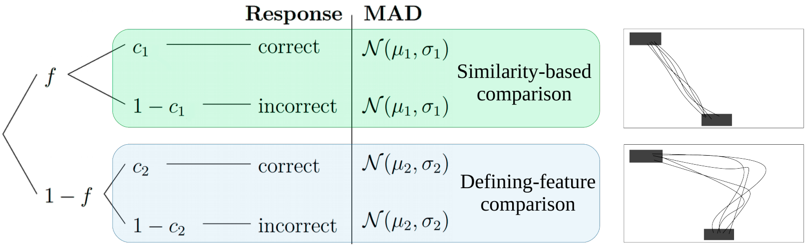
- The theory assumes two different processes
- f = probability of Process 1 (similarity-based comparison)
- c_1 = accuracy of similarity-based comparison
- c_2 = accuracy of defining-feature comparison
- With discrete responses only, the (MPT) model is not identifiable
  - only 1 free category for 3 free parameters



Identifiability of GPT Models

Identifiability of the feature comparison model

- Solution: Assume Gaussian component distributions for continuous variable
    - MAD = maximum absolute deviation in mouse-tracking
- Order constraint for mean parameters: $\mu_c < \mu_d$
    - Interpretation: More direct trajectories/small MADs for similarity-based comparison
- With both discrete *and* continuous data, model is identifiable
    - The different component distribution allow to disentangle the two processes



Flexibility: GPT Model Specification

GPTs as general-purpose measurement models

- GPTs can be specified as easily as MPTs
  - Implemented in the R package gpt
  - Currently under development: https://github.com/danheck/gpt
  - Define model in a text file similar to EQN
  - Type of latent distribution(s) defined within R (e.g., `latent="normal"`)

GPT version of 2-high-threshold model

```
# Tree ; Categ. ; MPT equation ; mean, SD (normal distr.)
target ; hit    ;  d           ; m_d,  sig
target ; hit    ; (1-d)*g      ; m_g,  sig
target ; miss   ; (1-d)*(1-g)  ; m_g,  sig

lure   ; cr     ;  d           ; m_d,  sig
lure   ; fa     ; (1-d)*g      ; m_g,  sig
lure   ; cr     ; (1-d)*(1-g)  ; m_g,  sig
```

Illustration of the gpt Package

```r
library("gpt")
# data from 2(response bias) x 2(memory strength) design:
# labels: "o30s_cr" = 30% old items / strong memory / correct rejection
head(heck2016, 3)
```

```
##       cat   rt
## 1 o30s_cr 1123
## 2 o30s_cr  671
## 3 o30s_cr  728
```

```r
modelfile <- "models/2htm_exgauss_2x2.txt"
# first lines:
```

```
## # 30% old / strong memory
## lure_s30;     o30s_cr   ;  (1-dn_s)*(1-g30) ; mu,sig,lambda_g_new30
## lure_s30;     o30s_cr   ;  dn_s             ; mu,sig,lambda_dn_s
## lure_s30;     o30s_fa   ;  (1-dn_s)*g30     ; mu,sig,lambda_g_old30
##
## target_s30;   o30s_hit  ;   do_s            ; mu,sig,lambda_do_s
## target_s30;   o30s_hit  ;  (1-do_s)*g30     ; mu,sig,lambda_g_old30
## target_s30;   o30s_miss ;  (1-do_s)*(1-g30) ; mu,sig,lambda_g_new30
##
## # 30% old / weak memory
## lure_w30;     o30w_cr   ;  (1-dn_w)*(1-g30)  ; mu,sig,lambda_g_new30
## lure_w30;     o30w_cr   ;  dn_w              ; mu,sig,lambda_dn_w
## lure_w30;     o30w_fa   ;  (1-dn_w)*g30      ; mu,sig,lambda_g_old30
```

gpt Package: Model Fitting

```
fit <- gpt_fit(x = "cat",          # MPT category
               y = "rt",           # name of continuous variable(s)
               data = heck2016,    # example data for 1 person
               file = modelfile,   # GPT model file
               latent="exgauss",   # family of latent RT distributions
               restrictions=list("dn_s=do_s", "dn_w=do_w"))
fit
```

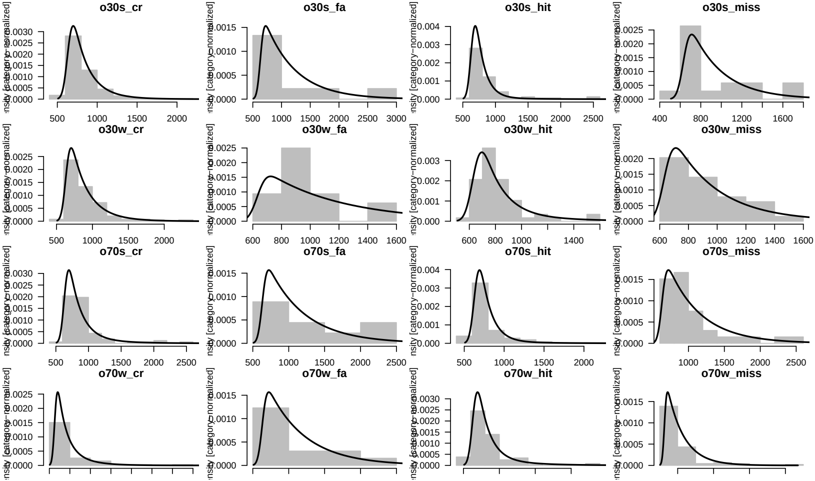
```
##                  Estimate     SE CI.lower CI.upper
## dn_s                0.741  0.027    0.687    0.794
## dn_w                0.477  0.033    0.411    0.542
## g30                 0.189  0.031    0.128    0.250
## g70                 0.261  0.038    0.186    0.335
## lambda_dn_s       172.252 18.192  136.597  207.907
## lambda_dn_w       192.571 35.080  123.815  261.327
## lambda_do_s       128.099 14.723   99.244  156.955
## lambda_do_w       151.588 18.455  115.417  187.760
## lambda_g_new30    311.057 30.895  250.504  371.610
## lambda_g_new70    460.131 36.179  389.221  531.042
## lambda_g_old30    531.286 86.541  361.669  700.903
## lambda_g_old70    516.769 74.457  370.837  662.702
## mu                633.896  5.376  623.360  644.432
## sig                49.279  4.047   41.347   57.212
```

```
test_fit(fit, bins = 4)$test     # Dzhaparidze-Nikulin statistic
```

```
##   statistic df      p.value
## 1  83.46215 42 0.0001469729
```

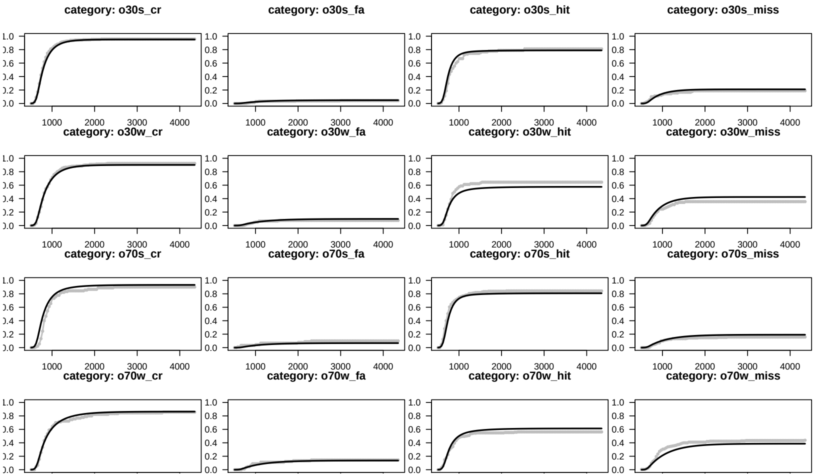
gpt Package: Model Fit

```r
hist(fit)
```



gpt Package: Model Fit

```r
plot(fit)   # cumulative densities
```



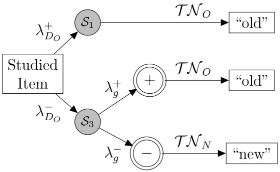
RT-MPT: Serial-process modeling of RTs

- Klauer, K. C., & Kellen, D. (2018). RT-MPTs: Process models for response-time distributions based on multinomial processing trees with applications to recognition memory. *Journal of Mathematical Psychology, 82*, 111–130.

RT-MPT

RT-MPT Models (Klauer & Kellen, 2018)

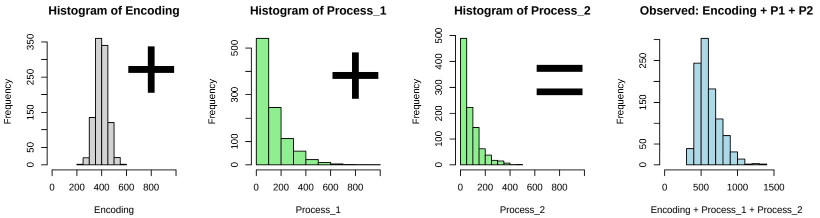
- Serial processing assumption: Observed RTs within each MPT branch are the result of a sequence of underlying processes
  1. Time for encoding and response execution
  2. Completion time for each state in the MPT model
  3. Observed RTs in a branch are the sum of encoding and all relevant processing times
- 2HTM components:
  - "Detection RTs": Encoding + Detection
  - "Guessing RTs": Encoding + unsuccessful detection + guessing



Parametric Assumptions

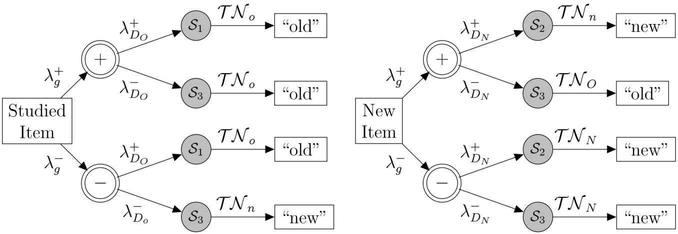
Illustration of the parametric assumptions

- Encoding and response execution (gray): truncated normal distribution (with mean μ and SD σ)
- Completion times (green): exponential distribution (with rate parameters λ)
- Observed time (blue): sum of encoding and processing times
- Note: Encoding and all completion times are independent



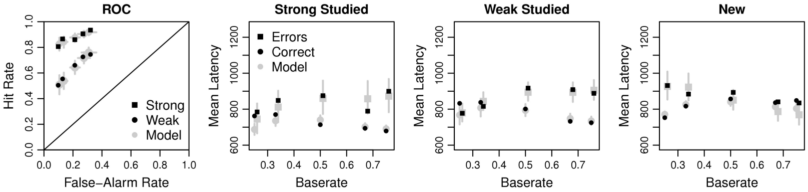
The ordering of the latent MPT states matters

- The 2HTM permits two possible orders:
  1. "Detect-Guess Model": First detection, then guessing (previous slides)
  2. "Default-Interventionist Model": First guessing (by default) and the detection process can intervent (see below)
- With the RT-MPT extension, both versions can be tested against each other



Empirical test of different 2HTM versions

- Bayesian hierarchical model (person random effects)
- The Default-Interventionist model fits 5 data sets better
- Effect of manipulations
  - No effect of memory strength on completion time of detection
  - Faster completion times for guessing if response matches the response-bias-condition
- The model fit is satisfactory:



Summary

Summary & Conclusion

RT-Extended MPT Models

1. No assumptions about shape of latent distributions
2. RT-extended MPT models are also MPT models
3. Testing relative speed of processes (stochastic dominance)

GPT Models

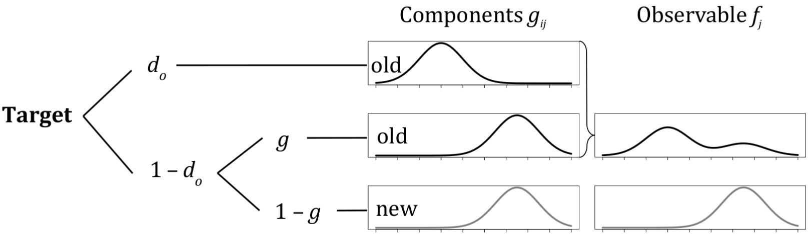
1. General approach for modeling discrete and continuous data
2. New tests of psychological theories (e.g., mouse-tracking)
3. MPT parameters estimated more precisely
4. User-friendly software (R package gpt)

RT-MPT

1. Assumption of serial processing (sum of encoding and completion times)
2. Hierarchical Bayesian

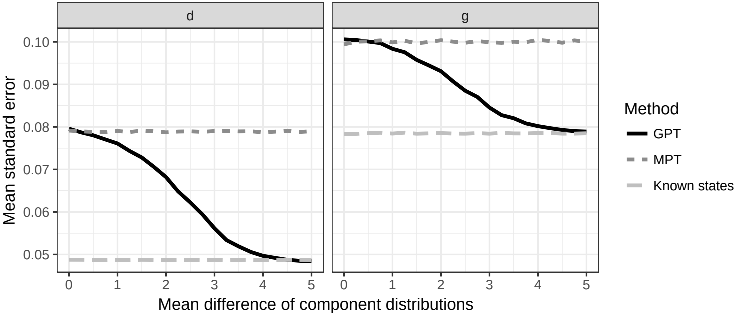
Appendix

- Higher precision of parameter estimates $\hat{\boldsymbol{\theta}}$
  - The more distinct the latent distributions, the smaller the standard error of $\hat{\boldsymbol{\theta}}$
  - Intuition: Continuous variables improve the classification of trials to latent processing states
  - Upper bound: Precision of MPT model
  - Lower bound: Latent states known



Appendix: GPT Estimates are More Precise

- Simulation of the 2HTM with Gaussian component distributions
  - $\mu^{\text{detect}} = 0$ vs. $\mu^{\text{guess}} = 0, \ldots, 5$
- Results
  - More distinct distributions: smaller standard error of $\hat{\boldsymbol{\theta}}$
  - Upper bound: Precision of MPT model
  - Lower bound: Latent states known



Appendix: Formal Definition of GPT Models

For a vector of discrete responses $\boldsymbol{x}$, a matrix of continuous variables $\boldsymbol{Y}$:

- The **joint distribution** f is a finite mixture
  - For a discrete response x and continuous response(s) $\boldsymbol{y}$:

$$f(x, \boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\eta}) = \sum_{j=1}^{J} \delta_{C_j}(\{x\}) \sum_{i=1}^{I_j} p_{ij}(\boldsymbol{\theta}) g_{ij}(\boldsymbol{y} \mid \boldsymbol{\eta})$$

- **Mixture weights** $p_{ij}(\boldsymbol{\theta})$
  - Identical to MPT-branch probabilities (probability parameters $\boldsymbol{\theta}$)

$$p_{ij}(\boldsymbol{\theta}) = c_{ij} \prod_{s=1}^{S} \theta_s^{a_{ijs}} (1 - \theta_s)^{b_{ijs}}$$

- **Basis distributions** $g_{ij}(\boldsymbol{y} \mid \boldsymbol{\eta})$ for latent states
  - E.g., normal, exGaussian, exWald,... distributions
  - Product-distributions for multivariate continuous data

Appendix: Formal Definition of GPT Models

- GPT models are a set of parameterized distributions:

$$\mathcal{M}^{\mathsf{GPT}}(\Theta = [0,1]^{S_1}, \Lambda \subset \mathbb{R}^{S_2}) = \{f(x, \boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\eta}) \mid \boldsymbol{\theta} \in \Theta, \boldsymbol{\eta} \in \Lambda\}$$

- Likelihood where trials $k = 1, \ldots, K$ fall into disjoint sets M_t that are modeled by $t = 1, \ldots, T$ processing trees

$$L(\boldsymbol{\theta}, \boldsymbol{\eta} \mid \boldsymbol{x}, \boldsymbol{Y}) = \prod_{t=1}^{T} \prod_{k \in M_t} f_t(x_k, \boldsymbol{y}_k \mid \boldsymbol{\theta}, \boldsymbol{\eta})$$

Appendix: GPT Parameter Estimation

Expectation-Maximization (EM) algorithm

- E: Compute expected probabilities $\boldsymbol{z}$ of being in the cognitive states
  - Continuous variables inform the state-vector $\boldsymbol{z}$
- M: Maximize likelihood of continuous parameters given the latent-states $\boldsymbol{z}$

Illustration

- E-step estimates the probability to be in state i in trial k:

$$P(z_k = i \mid \boldsymbol{\theta}, \boldsymbol{\eta}, x_k, \boldsymbol{y}_k) = \frac{P(z_k = i \mid \boldsymbol{\theta}, x_k) g_{ij}(\boldsymbol{y}_k \mid \boldsymbol{\eta})}{\sum_i P(z_k = i \mid \boldsymbol{\theta}, x_k) g_{ij}(\boldsymbol{y}_k \mid \boldsymbol{\eta})}$$

Cat.	RT [ms]	Conf. [1-10]	ERP [mV]	$z_k = 1$	...	$z_k = I$
c_1	551	3	1.324	0.43	...	0.09
c_1	502	1	0.921	0.19	...	0.56
c_2	470	6	2.231	0.30	...	0.00
c_1	733	4	1.010	0.14	...	0.47

Appendix: Identifiability

Identifiability: GPT models with identifiable MPT structure are identifiable if (cf. distribution-free approach):

- Component distributions are observable (Yantis et al., 1991)
- Stepwise deletion of identifiable component distributions (Heck & Erdfelder, 2016)
- Specific matrix has full rank (Heck & Erdfelder, 2016)

Alternative strategy

- Using order constraints to identify GPT models with a nonidentifiable MPT structure
- Label switching of processing paths with component distributions